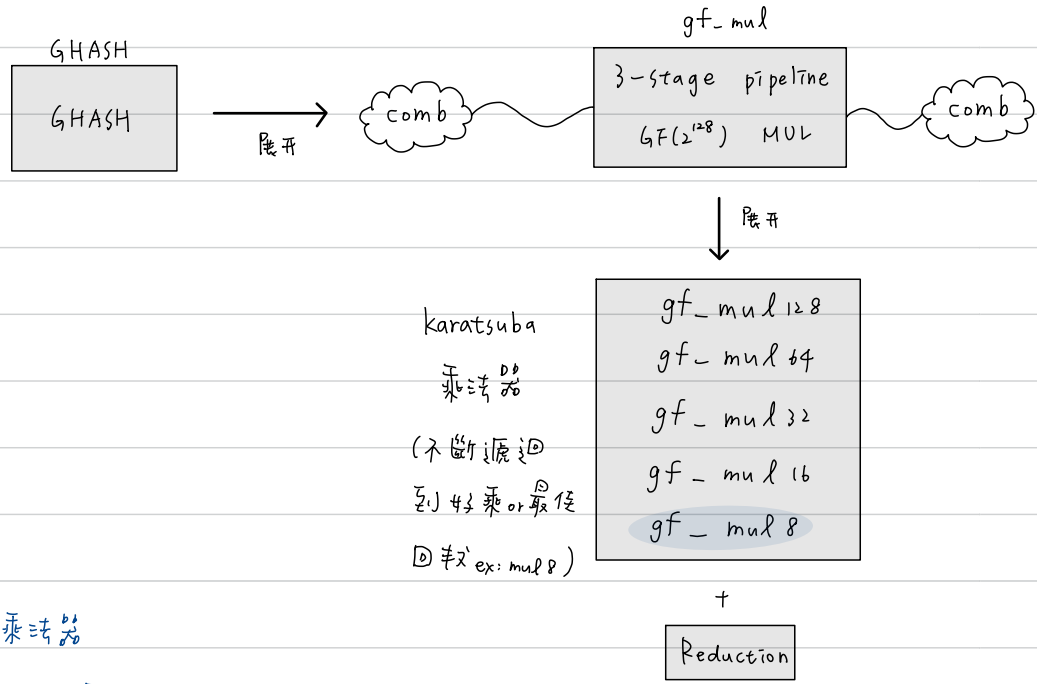
# GHASH module




乘法器

<一般> 看下一張

<二元有限域 $GF(2^8)$> <無 modulo，最后会 reduction>

ex: {57}·{83} = out (初始設 00000000)

(01010111)·(10000011)

$\Rightarrow (x^6+x^4+x^2+x+1)\cdot(x^7+x+1)$

$= [(x^6+x^4+x^2+x+1)\cdot x]^{(a)} +$ $b[0]=1$, out = (a<<0) ^ out = 01010111

$[(x^6+x^4+x^2+x+1)\cdot x^2]^{(a)} +$ $b[1]=1$, out = (a<<1) ^ (01010111) = 10101110 ^ 01010111 = 11111001

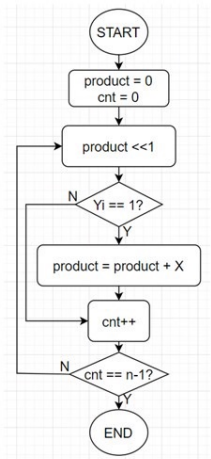
$[(x^6+x^4+x^2+x+1)\cdot x^7]$ $b[7]=1$, out = (a<<7) ^ (11111001) = 1010111100000000 ^ 11111001 = 1010111011111001

$= x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1$

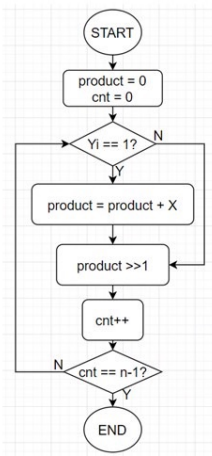
# N-bit*N-bit Multiplier based on N-bit Adder

Ex: X=1111, Y=0101

一开始左移，为了程式好看




| START | 00000000 |
|---|---|
| << 1 | 00000000 |
| $Y_3$ == 0 | 00000000 |
| << 1 | 00000000 |
| $Y_2$ == 1 | +1111 |
| | 00001111 |
| << 1 | 00011110 |
| $Y_1$ == 0 | 00011110 |
| << 1 | 00111100 |
| $Y_0$ == 1 | +1111 |
| | 01001011 |
| END | |




| START | | 00000000 |
|---|---|---|
| $Y_0$ == 1 | + | 1111 |
| | | 11110000 |
| >> 1 | | 01111000 |
| $Y_1$ == 0 | | 01111000 |
| >> 1 | | 00111100 |
| $Y_2$ == 1 | + | 1111 |
| | 1 | 00101100 |
| >> 1 | | 10010110 |
| $Y_3$ == 0 | | 10010110 |
| >> 1 | | 01001011 |
| END | | |

(乘1) (乘0)

* 又需看 y 是因為二進制(只有 1, 0)，所以如果 $y_i$=1 則加x並移位，$y_i$=0 又需移位

* 移位？

ex: 〈左移〉 << 1 跟平常運算相似

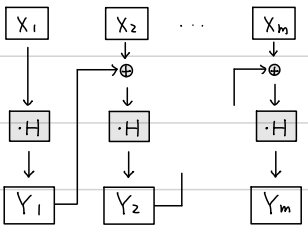
〈右移〉 >> 1

```
y3 start        125 (x)
y2              x326 (y)
y1              750
                250
<<1 <<1         375 ←
<<1 <<1 <<1    40750
```

```
125 (x)
x326 (y)
750
250
375
40750
```

```
y3 start   000000
           +  375
y2    <<1  003750
           +  250
y3    <<1  040000
           +  750
           40750 #
```

```
y1 start   000000
           +  750
>>1        075000
y2         +  250
>>1        032500
           +  375
>>1        040750 #
```




* 每次的乘加運算都需要輸入上一次運算結果

# AES演算法

128 bits 組成一個狀態(state)，state 為 AES 回合運算的基本單位。key 長度不同，分為 AES-128、AES-196、AES-256，所需回合運算也不同

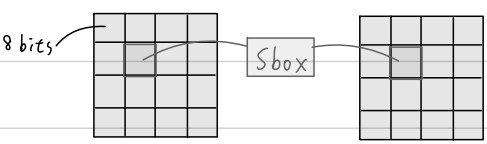
| | 密鑰長度(32 bits) (128, 192, 256) | 分組長度(32 bits) (128) | 加密輪數 (10-14輪) |
|---|---|---|---|
| AES-128 | 4 | 4 | 10 |
| AES-192 | 6 | 4 | 12 |
| AES-256 | 8 | 4 | 14 |

$a_0\ a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8\ a_9\ a_{10}\ a_{11}\ a_{12}\ a_{13}\ a_{14}\ a_{15}$

| | | | |
|---|---|---|---|
| $a_0$ | $a_4$ | $a_8$ | $a_{12}$ |
| $a_1$ | $a_5$ | $a_9$ | $a_{13}$ |
| $a_2$ | $a_6$ | $a_{10}$ | $a_{14}$ |
| $a_3$ | $a_7$ | $a_{11}$ | $a_{15}$ |

| | | | |
|---|---|---|---|
| 127-120 | 95-88 | 63-56 | 31-24 |
| 119-112 | 87-80 | 55-48 | 23-16 |
| 111-104 | 79-72 | 47-40 | 15-8 |
| 103-96 | 71-64 | 39-32 | 7-0 |

## *SubBytes 組合 Sbox 轉換




Sbox 單純轉換(一格)

SubBytes / InvSubBytes 是全部轉換，呼叫 sbox

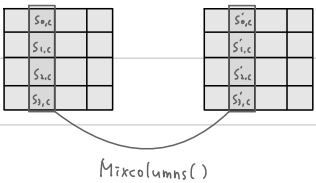
## *ShiftRows (列循環位移運算)

| | | | | |
|---|---|---|---|---|
| 127-120 | 95-88 | 63-56 | 31-24 | X |
| 119-112 | 87-80 | 55-48 | 23-16 | 左 1 |
| 111-104 | 79-72 | 47-40 | 15-8 | 左 2 |
| 103-96 | 71-64 | 39-32 | 7-0 | 左 3 |

## *InvShiftRows

| | | | | |
|---|---|---|---|---|
| 127-120 | 95-88 | 63-56 | 31-24 | X |
| 119-112 | 87-80 | 55-48 | 23-16 | 右 1 |
| 111-104 | 79-72 | 47-40 | 15-8 | 右 2 |
| 103-96 | 71-64 | 39-32 | 7-0 | 右 3 |

## ✳ MixColumns (混行轉換) & InvMixColumns (反轉換)




$S'(x) = a(x) \otimes S(x)$

$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$
(0b 0d 09 0e)

GF(2)

* mul 1 (00000001)：自己
* mul 2 (00000010)：左移 <<1，若最高位1 ⇒ overflow，(00000001 → 00000010) modulo 8'h1b (00011011)
* mul 3 (00000011)：mul2 ⊕ 自己

沒有⊕這位是因為只有八位！

(取餘數) ⊕ $m(x) = x^8 + x^4 + x^3 + x + 1$

(=元素的 mod = ⊕)

ex: {57}·{83} = {c1}

$(x^6 + x^4 + x^2 + x + 1)(x^7 + x + 1)$

(01010111)(10000011)

mul2的7次 ⊕ mul2的1次 ⊕ 自己

### MixColumns & InvMixColumns

$$\begin{bmatrix} S'_0 \\ S'_1 \\ S'_2 \\ S'_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{bmatrix}$$

$$\begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix}$$

$S'_{0,c} = (\{02\} \cdot S_{0,c}) \oplus (\{03\} \cdot S_{1,c}) \oplus S_{2,c} \oplus S_{3,c}$

$S'_{1,c} = S_{0,c} \oplus (\{02\} \cdot S_{1,c}) \oplus (\{03\} \cdot S_{2,c}) \oplus S_{3,c}$

$S'_{2,c} = S_{0,c} \oplus S_{1,c} \oplus (\{02\} \cdot S_{2,c}) \oplus (\{03\} \cdot S_{3,c})$

$S'_{3,c} = (\{03\} \cdot S_{0,c}) \oplus S_{1,c} \oplus S_{2,c} \oplus (\{02\} \cdot S_{3,c})$

① $\Rightarrow (x^6 + x^4 + x^2 + x + 1) \cdot x = x^7 + x^5 + x^3 + x^2 + x$ (<<1)

② $(x^7 + x^5 + x^3 + x^2 + x) \cdot x = x^8 + x^6 + x^4 + x^3 + x^2 \bmod x^8 + x^4 + x^3 + x + 1$
$= x^6 + x^2 + x + 1$ (<<1)(再⊕1b)

③ $(x^6 + x^2 + x + 1) \cdot x = x^7 + x^3 + x^2 + x$

④ $(x^7 + x^3 + x^2 + x) \cdot x = x^8 + x^4 + x^3 + x^2 \bmod x^8 + x^4 + x^3 + x + 1$
$= x^2 + x + 1$ (<<1)(再⊕1b)

⑤ $(x^2 + x + 1) \cdot x = x^3 + x^2 + x$ (<<1)

⑥ $(x^3 + x^2 + x) \cdot x = x^4 + x^3 + x^2$ (<<1)

⑦ $(x^4 + x^3 + x^2) \cdot x = x^5 + x^4 + x^3$ (<<1)

⑧ $(x^6 + x^4 + x^2 + x + 1) \cdot x = x^7 + x^5 + x^3 + x^2 + x$ (<<1)

total (mul2 7次)⊕(mul2 2次)⊕自己

$(x^5 + x^4 + x^3) \oplus (x^7 + x^5 + x^3 + x^2 + x) \oplus (x^6 + x^4 + x^2 + x + 1) = x^7 + x^6 + 1$ #

→ 3個 function

(這邊可簡在02,03 function裡)(09,0b,0d,0e需重複呼叫才多出 multiply function)

① ~~乘!的 function (multiply)~~

用來呼叫的 function ② 02 ③ 03

(為了程式碼好看，多寫1個 01 function)

→ 5個 functions

① 乘！的 function (multiply)

用來呼叫的 function ② 09 (1001) ⇒ {02} 3次 ⊕ 自己
③ 0b (1011) ⇒ {02} 3次 ⊕ {02} ⊕ 自己
④ 0d (1101) ⇒ {02} 3次 ⊕ {02} 2次 ⊕ 自己
⑤ 0e (1110) ⇒ {02} 3次 ⊕ {02} 2次 ⊕ {02}

## * AddRoundKey

in ⊕ key

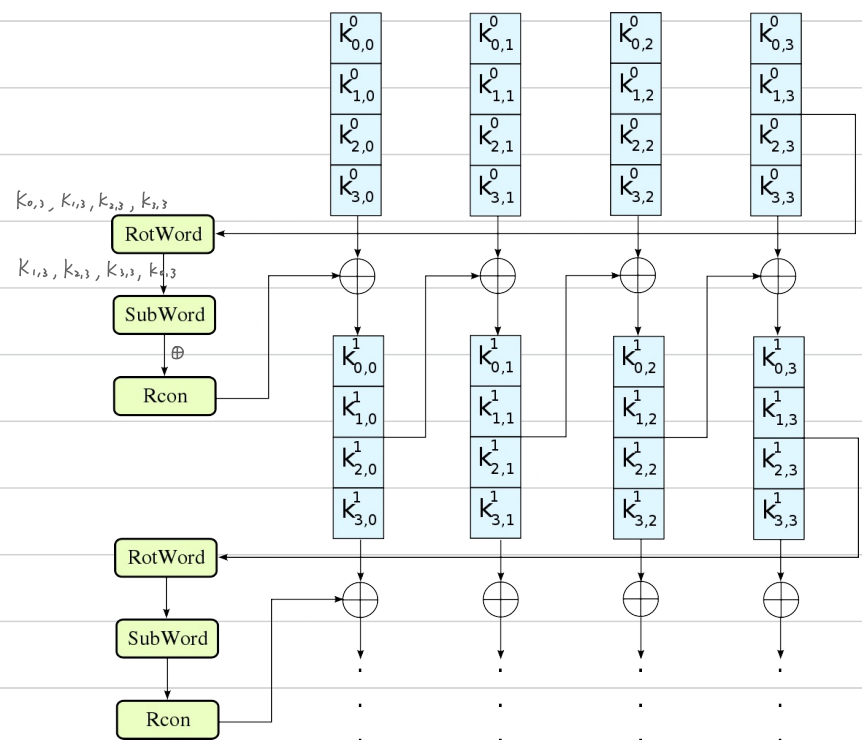
## * KeyExpansion

⟶ Rotword : w[B0, B1, B2, B3] → Rotword → w[B1, B2, B3, B0]

⟶ Subword : 用 Sbox 轉換 (Sbox 一次 8 bits 轉換 ⇒ 4次)

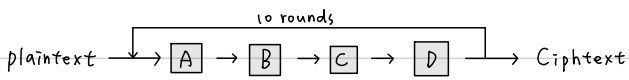
⟶ Rcon : 輸入回合數, 得該回合的回合常數 (对照表)

| | | | |
|---|---|---|---|
| 127-120 | 95-88 | 63-56 | 31-24 |
| 119-112 | 87-80 | 55-48 | 23-16 |
| 111-104 | 79-72 | 47-40 | 15-8 |
| 103-96 | 71-64 | 39-32 | 7-0 |

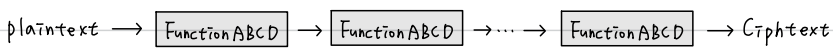


AES (top module)


* 我的作法 (area 小)



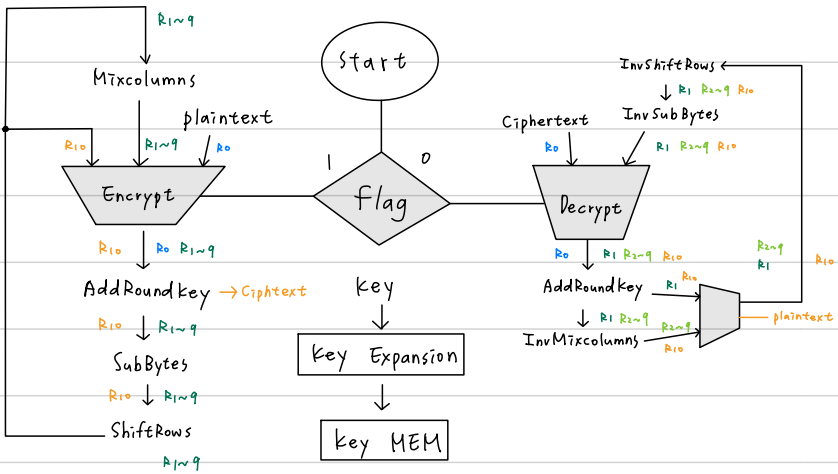

* 任華的作法 (throughput 高) (area 為我的 10 倍)

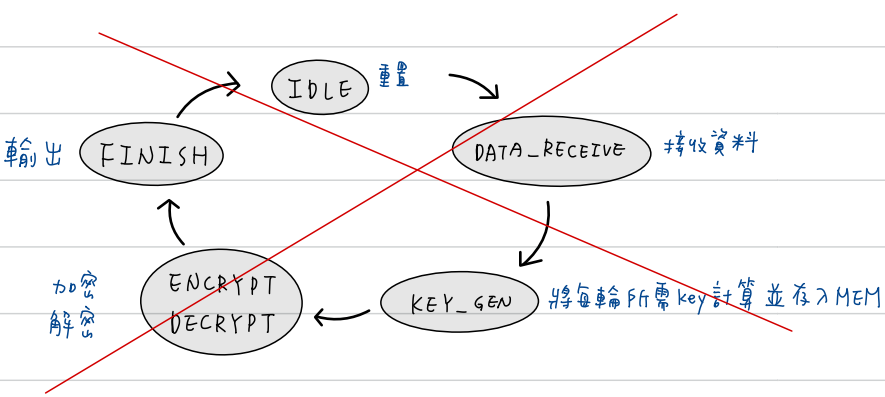



* 兩個的 latency (延遲) same ⇒ 輸出第一筆 Ciphtext 時間 same

但任華的 throughput 高 ∵ 假設第一筆輸出需 10 clk, 則 20 clk 時, 我的只有 2 个輸出, 任華有 11 个 (2 倍)
(10 倍)

* 任華 aes 只有加密!


Start
flag
1
0
Mixcolumns
R1~9
plaintext
R10
R1~9
R0
Encrypt
R10
R0 R1~9
AddRoundkey
Ciphtext
R10
R1~9
SubBytes
R10
R1~9
ShiftRows
R1~9
key
Key Expansion
Key MEM
InvShiftRows
R1 R2~9 R10
InvSubBytes
Ciphertext
R0
R1 R2~9 R10
Decrypt
R0
R1 R2~9 R10
AddRoundkey
R10
R1
R1 R2~9
InvMixcolumns
R2~9
R10
R2~9
R1
R10
plaintext


IDLE
重置
DATA_RECEIVE
接收資料
KEY_GEN
將每輪所需key計算並存入MEM
ENCRYPT
DECRYPT
加密
解密
FINISH
輸出